WILD 502
Lab 7 - Closed Population C-R Models

Today, you will (1) build a variety of closed population models in MARK, (2) check estimates from non-likelihood based models in CAPTURE, and (3) compare multiple groups. For those of you who go on to use closed population C-R models, you'll want to check out the chapter on the topic in *C&W* and investigate the mixture models and the Huggins models as well, which are beyond what we have time for this semester.

Building the basic models in MARK
To get started, you'll first work with an artificial data set with 1 group and $t = 6$ that analyzes the data in *ClosedSim.inp*, which is available on the course schedule for lab 7.

M(0): constant *p; & p = c*

The 3 PIMs appear as follows for (1) *p*, (2) *c*, & (3) *f0*:

Real Function Parameters of {M(0)}

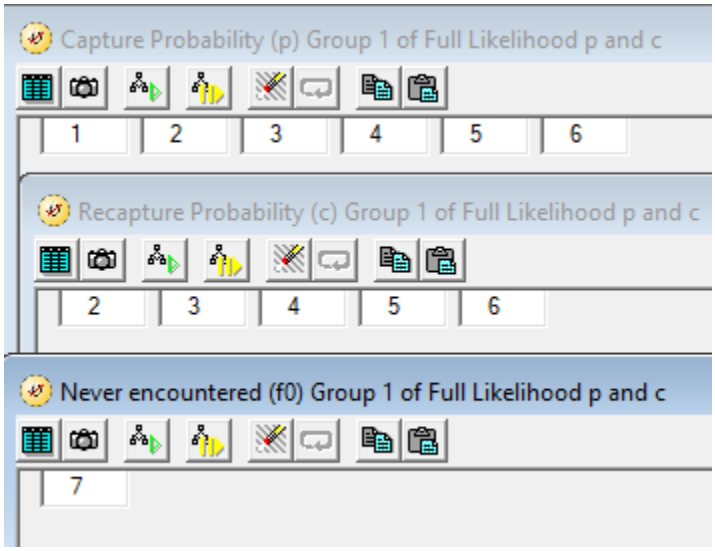
Parameter	Estimate	Standard Error	95% Confidence Interval Lower	Upper
1:p	0.2631441	0.0114442	0.2413354	0.2861801
2:f0	66.388583	11.606446	47.249500	93.280226

Look at the full model output (click on the icon to the right of the trash can) and find the value for M_{t+1}, which is 351. Your estimate of *f0* is 66.39. So your estimate of *N* is 351 + 66.39 or 417.39 with SE = SE(*f0*) = 11.61. You can check this by looking at the output for derived parameters (4th icon to the right of the trash can or use output menu for specific model output), where the ouput for *N* is presented for closed models.

Estimates of Derived Parameters
Population Estimates of {M(0)}

Grp.	Sess.	N-hat	Standard Error	95% Confidence Interval Lower	Upper
1	1	417.38858	11.606446	398.24950	444.28023

M(t): p varies by occasion & $p = c$.



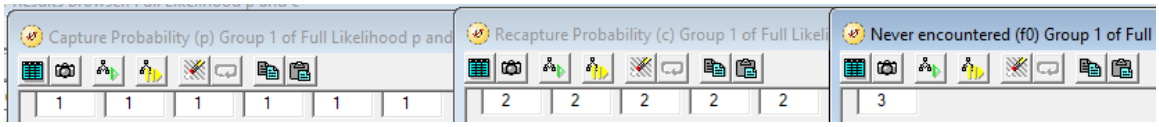
Real Function Parameters of {M(t)}

Parameter	Estimate	Standard Error	95% Confidence Interval Lower	Upper
1:p	0.2494707	0.0223286	0.2083046	0.2957333
2:p	0.2276723	0.0215273	0.1882442	0.2725856
3:p	0.1695432	0.0190220	0.1354430	0.2101420
4:p	0.4165919	0.0267273	0.3653263	0.4697272
5:p	0.2373605	0.0218916	0.1971446	0.2828894
6:p	0.2954896	0.0238211	0.2510259	0.3442096
7:f0	61.874070	11.111034	43.637777	87.731337

Estimates of Derived Parameters
Population Estimates of {M(t)}

Grp.	Sess.	N-hat	Standard Error	95% Confidence Interval Lower	Upper
1	1	412.87407	11.111034	394.63778	438.73134

M(b): $p \neq c$ but p and c each constant over time.



Real Function Parameters of {M(b)}

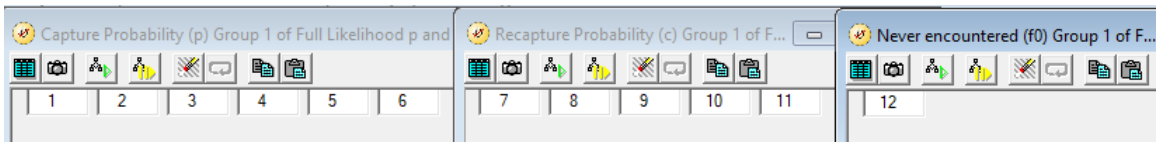
Parameter	Estimate	95% Confidence Interval Standard Error	Lower	Upper
1:p	0.1775135	0.0266127	0.1311823	0.2357675
2:c	0.2878505	0.0138413	0.2614998	0.3157213
3:f0	156.88576	46.759823	88.563877	277.91401

Estimates of Derived Parameters
Population Estimates of {M(b)}

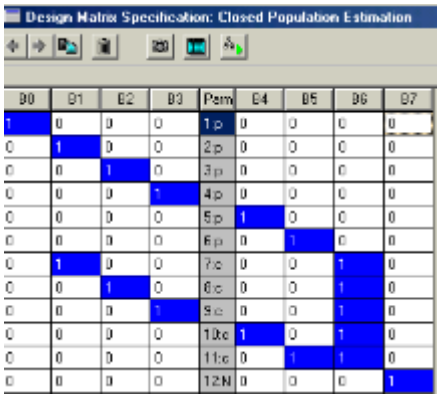
Grp.	Sess.	N-hat	Standard Error	95% Confidence Interval Lower	Upper
1	1	507.88576	46.759823	439.56388	628.91401

Notice how the abundance estimate is higher for this model: it appears that recapture rates are higher than capture rates for unmarked animals (evidence of trap happiness). You're trying to estimate how many unmarked animals were never caught (*f0*) and so the lower capture rate *p* causes the estimate of the number that went uncaptured to be higher.

M(tb - additive)



This seems easy enough, but … under M(tb - additive) we need some additional constraint on p_i and c_i to avoid over-parameterizing the model. One way to handle this is to model a constant difference between the log-odds of *p* and the log-odds of *c*. Thus, occasions with higher *p* also have higher *c*. For this type effect, we can use the design matrix to further *constrain* the estimates.



Look this DM over carefully. Notice that (1) $p_2 - p_6$ are constrained to be different by a constant amount (B6) from $c_2 - c_6$ and (2) there is no c_1. This is different from letting the *c* vary independently from the *p*, i.e., using B0-B5 for $p_1 - p_6$ and B6-B10 for $c_2 - c_6$.

Real Function Parameters of {M(tb- additive)}

Parameter	Estimate	Standard Error	95% Confidence Interval Lower	Upper
1:p	0.1519343	0.0596456	0.0674331	0.3074190
2:p	0.1087197	0.0542612	0.0391118	0.2676968
3:p	0.0689114	0.0390369	0.0219668	0.1960675
4:p	0.1773005	0.0968699	0.0553889	0.4419882
5:p	0.0778962	0.0496742	0.0213160	0.2467896
6:p	0.0960303	0.0616427	0.0257326	0.2993623
7:c	0.3057709	0.0366060	0.2390380	0.3817867
8:c	0.2108827	0.0255312	0.1651606	0.2652408
9:c	0.4376205	0.0268714	0.3858412	0.4907955
10:c	0.2337315	0.0214465	0.1943387	0.2783505
11:c	0.2772364	0.0234123	0.2337562	0.3253700
12:f0	326.92435	258.92915	83.267173	1283.5734

Estimates of Derived Parameters

Population Estimates of {M(tb- additive)}

Grp.	Sess.	N-hat	Standard Error	95% Confidence Interval Lower	Upper
1	1	677.92435	258.92915	434.26717	1634.5734

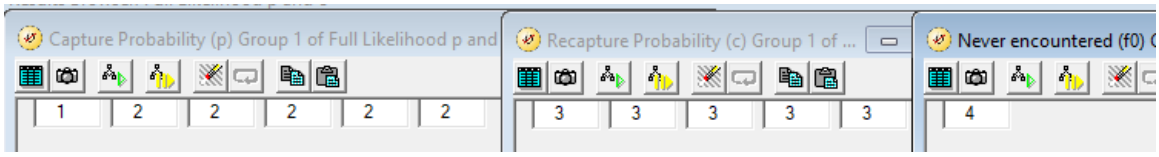
Week	*p*	*c*	Diff
1	0.152		(c - p)
2	0.109	0.306	0.197
3	0.069	0.211	0.142
4	0.177	0.438	0.260
5	0.078	0.234	0.156
6	0.096	0.277	0.181

Clearly, the difference between p_i & c_i is ***not*** constant: remember that the difference is only constant on log-odds scale (see the 7th beta-hat; all the log-odds for c_i are higher by ~1.28 compared to the log-odds of p_i for the same occasion, e.g., in R, type the following: `qlogis(0.306) - qlogis(0.109)`, which yields 1.28) When we back-transform to the real parameters, the difference is no longer constant.

M(bh): Multiple versions

The simplest version is M(b) or $\bar{p}_1 = \bar{p}_2 = \bar{p}_3 = ...\bar{p}_t$ - we've already got that one.

The next version, let's call it (M(bh-2p's), estimates $\bar{p}_1 \neq \bar{p}_2 = \bar{p}_3 = ...\bar{p}_t$



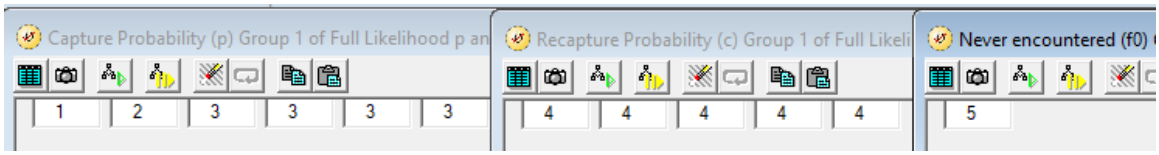
Real Function Parameters of {M(bh - 2 p's)}

Parameter	Estimate	Standard Error	95% Confidence Interval Lower	Upper
1:p	0.1625518	0.0386470	0.1001275	0.2529542
2:p	0.1181953	0.0398891	0.0595376	0.2210598
3:c	0.2878505	0.0138413	0.2614998	0.3157213
4:f0	282.64438	139.39522	113.25718	705.36672

Population Estimates of {M(bh - 2 p's)}

Grp.	Sess.	N-hat	Standard Error	95% Confidence Interval Lower	Upper
1	1	633.64438	139.39522	464.25718	1056.3667

The next version, call is (M(bh-3p), estimates $\bar{p}_1 \neq \bar{p}_2 \neq \bar{p}_3 = ...\bar{p}_t$



Real Function Parameters of {M(bh - 3 p's)}

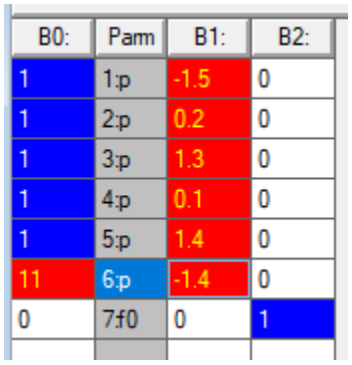
Parameter	Estimate	Standard Error	95% Confidence Interval Lower	Upper
1:p	0.1737641	0.0459635	0.1009541	0.2825810
2:p	0.1245514	0.0403775	0.0644155	0.2271947
3:p	0.1331880	0.0567101	0.0554179	0.2869423
4:c	0.2878505	0.0138413	0.2614998	0.3157213
5:f0	241.75770	147.53267	80.251040	728.29944

Population Estimates of {M(bh - 3 p's)}

Grp.	Sess.	N-hat	Standard Error	95% Confidence Interval Lower	Upper
1	1	592.75770	147.53267	431.25104	1079.2994

In these 2 versions of M(bh), notice how the estimated p_i values are decreasing (you're catching the easier-to-catch animals at higher rates than the harder-to-catch animals such that as you progress through the occasions the remaining unmarked animals tend to be those that are harder to catch).

You can keep going out 1 or 2 more if ΔAIC values indicate this is appropriate. But, beyond this you can't build any more of the classic or canned closed-capture models in MARK, i.e., you can't work with M(th) or M(tbh). You can, however, incorporate group covariates if you want. For example, you might have some measure of weather for each week and want to model *p* and *c* as a function of weather. For example, let's say you had average air temperatures for each night of trapping (standardized to a mean of 0 and SD of 1). You could estimate using this information. Start by recalling M(t) and then call up a design matrix with t3 columns. You can enter the standardized temperature data as shown below and run the model calling it M(temp).



B0:	Parm	B1:	B2:
1	1:p	-1.5	0
1	2:p	0.2	0
1	3:p	1.3	0
1	4:p	0.1	0
1	5:p	1.4	0
11	6:p	-1.4	0
0	7:f0	0	1

M(th): use CAPTURE (available from MARK's "Tests" menu) to check the estimates from this model. The output is:

Population estimate under time variation and individual heterogeneity in capture probabilies.
See model M(th) of Chao et al. (1992).

Number of trapping occasions was 6
Number of animals captured, M(t+1), was 351
Total number of captures, n., was 659

Frequencies of capture, f(i)

i=	1	2	3	4	5	6
f(i)=	153	120	53	18	7	0

Estimator	Gamma	N-hat	se(N-hat)
1	0.1741	491.82	24.43
2	0.0724	430.70	19.78
3	0.1041	449.36	21.70

p-hat(j) = 0.23 0.21 0.16 0.38 0.22 0.27

Bias-corrected population estimate is 449 with standard error 21.7002
Approximate 95 percent confidence interval 416 to 501

M(h): use CAPTURE to check jackknife estimates from this model. The output is:

Number of trapping occasions was 6
Number of animals captured, M(t+1), was 351
Total number of captures, n., was 659

Frequencies of capture, f(i)

i=	1	2	3	4	5	6
f(i)=	153	120	53	18	7	0

Computed jackknife coefficients

	N(1)	N(2)	N(3)	N(4)	N(5)
1	1.833	2.500	3.000	3.333	3.500
2	1.000	0.467	-0.233	-0.833	-1.167
3	1.000	1.000	1.225	1.542	1.750
4	1.000	1.000	1.000	0.956	0.914
5	1.000	1.000	1.000	1.000	1.001

The results of the jackknife computations

i	N(i)	SE(i)	.95 Conf. Limits		Test of N(i+1) vs. N(i)
0	351				Chi-square (1 d.f.)
1	478.5	15.29	448.5	508.5	14.690
2	516.5	23.32	470.8	562.2	0.196
3	520.9	31.10	460.0	581.9	0.383
4	515.9	37.64	442.1	589.7	0.885
5	511.7	41.36	430.7	592.8	0.000

Average p-hat = 0.2166

Interpolated population estimate is 507 with standard error 24.7136
Approximate 95 percent confidence interval 466 to 563

Model Averaged Estimate of *N* from MARK

First, look at the MARK results browser for the models run:

Model	AICc	Delta AICc	AICc-wt	Model-Like	K	Deviance	-2log(L)
M(tb-ad)	-960.3660	0.0000	0.98698	1.00000	8	139.3618	-976.4347
M(t)	-951.7102	8.6558	0.01302	0.01320	7	150.0329	-965.7636
M(bh-2p)	-904.4809	55.8851	0.00000	0.00000	4	203.2966	-912.5000
M(bh-3p)	-902.6037	57.7623	0.00000	0.00000	5	203.1642	-912.6323
M(b)	-901.7478	58.6182	0.00000	0.00000	3	208.0373	-907.7592
M(0)	-889.2287	71.1373	0.00000	0.00000	2	222.5621	-893.2344
M(temp)	-888.6865	71.6795	0.00000	0.00000	3	221.0986	-894.6980

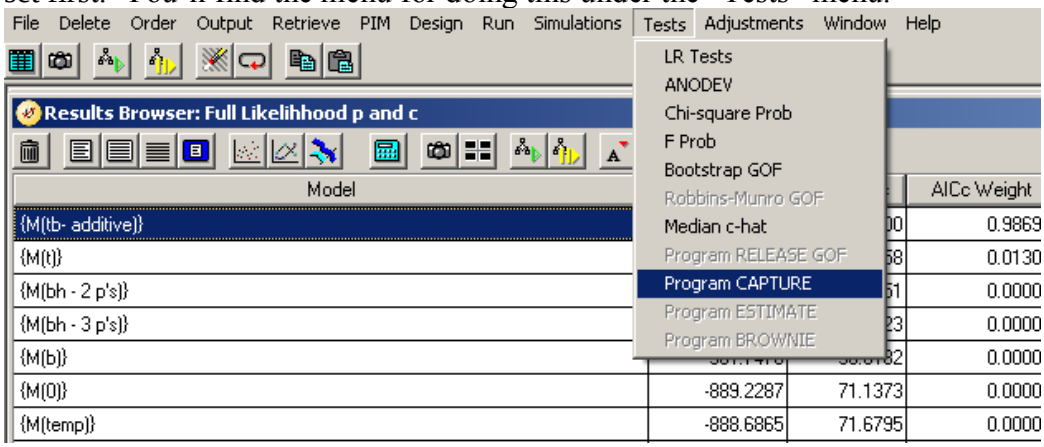
M(tb-additive) and M(t) dominate the model weights. So, they will dictate the average estimate of population that results from model averaging.

Population Size (N) Group 1 Parameter 12

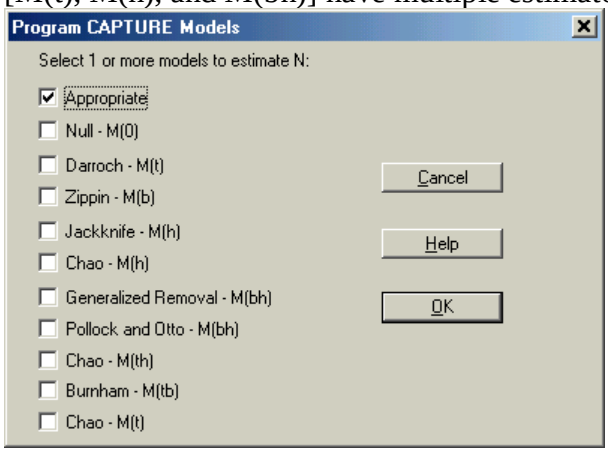
Model	Weight	Estimate	Standard Error
{M(tb- additive)}	0.98698	677.9243000	258.9281200
{M(t)}	0.01302	412.8740700	11.1110330
Weighted Average		674.4724496	255.7007034
Unconditional SE			258.9888819

95% CI for Weighted Average Estimate is 166.8542411 to 1182.0906582
Percent of Variation Attributable to Model Variation is 2.52%

GOF? Well, you can't test GOF for closed population models in MARK yet. So, like it or not (!), you have to use the output from Program CAPTURE. This can nicely be done within MARK but …. ONLY after you've built at least 1 model in MARK for the data set first. You'll find the menu for doing this under the "Tests" menu.



If you click on the "Program Capture" button, you obtain the following window, which as you can see has check boxes for ALL the models we've discussed (and some models [M(t), M(h), and M(bh)] have multiple estimators).



If you choose "Appropriate", then CAPTURE will check all possible models, run GOF tests, and attempt to choose the best model. We're not focused on model selection based on these tests though CAPTURE users used to be. We're more interested in seeing if models of interest fit the data. Although you might not have exactly the test you'd like here, you can look through the GOF tests and look for evidence of which model seem to fit (or not fit) the data. Finally, you can also take a look at how models M(h) and M(th), which we couldn't build in MARK perform in CAPTURE.

Assignment to work through in class but <u>do not need to turn in</u> given that you are working on the take-home exam:

Work with **capture.inp**, a simulated data set for which I know the truth. This data set is for $t = 7$ and for 1 group of animals. Thus, it's a pretty simple exercise to simply re-do what you've seen in the previous pages of this handout. With this in mind, you should:

1. Build and run the potential models in MARK as seen above – these are M(0), M(t), M(b), M(tb), M(bh-2p's), M(bh-3p's), M(bh-4p's), and M(bh-5p's).

2. Evaluate whether the bait type used was related to *p*. On occasions 1, 2, and 3, bait A was used and on occasions 4-7, bait B was used. Use the design matrix and dummy variables to:
   a. build a model that ignores time variation other than that due to bait.
   b. build a model that considers basic time variation plus effects of bait.

3. Conduct model averaging for N-hat and consider which models weighed most heavily on the average estimate. Does this estimate seem precise, useful? Be sure no models provide an abundance estimate equal to M_{t+1}; if they do, be sure all seems well with the estimation before using the estimate in model averaging.

4. Examine estimates for Chao's estimators for M(h) and M(th) from CAPTURE. Do these estimates differ much from what you obtained via model averaging?

5. Examine the test output from CAPTURE and see if it appears that you have relevant models that fit the data in your MARK model list. In some cases you get a GOF test for the model you chose in MARK, which is nice. Note: we would simply like to be able to test that our most general model fits the data but can't do this, i.e., we can't check the fit of M(tbh).

6. Consider how well your estimates inform about the true population: this data set came from a true generating model where N=200; p_i for t = 1-7 were 0.5, 0.3, 0.4, 0.5, 0.4, 0.3, and 0.4. So, what was the true model? Did you correctly identify this model for this data set, which is only a single simulation? Did your 95% CI for N-hat contain the true value?

In *Cooch & White,* the chapter on closed-population, capture-recapture models by Paul Lukacs is very helpful. It is especially at explaining many of the newer models. If you're using closed captures in your work, it's well worth spending time with that chapter, and, of course, with the primary literature and the Williams et al. text.